49658-0024

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:	Stewart SABADELL et al.		
Serial No.:	09/286,133	Examiner	Ayal I. Sharon
Filing Date:	April 1, 1999	Art Unit:	2123

For: TRANSLATING OBJECTS BETWEEN SOFTWARE APPLICATIONS WHICH EMPLOY DIFFERENT DATA FORMATS

Commissioner for Patents
Box 1450
Alexandria, VA 22313-1450

SUPPLEMENTAL APPEAL BRIEF

Sir:

This Supplemental Appeal Brief is submitted in support of the Notice of Appeal filed on August 21, 2002. The Office Action mailed March 10, 2003 (Paper No. 18) reopened prosecution on the merits. Reinstatement of the Appeal, based hereon, is requested.

I. REAL PARTY IN INTEREST

Autodesk, Inc. is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

To the present knowledge of Appellant and Appellant's legal representative, there are currently no related appeals or interference proceedings in progress which will directly affect, or be directly affected by, or have a bearing on the Board's decision in the present Appeal.

III. STATUS OF CLAIMS

Claims 1-14, 16 and 17 are pending in this Application. Claims 1, 2, 4-10 and 12-13 have been finally rejected under 35 U.S.C. §102(b) as allegedly anticipated by Lowry et al. (U.S. Patent No. 4,864,497; hereinafter "*Lowry*"). Claims 3 and 11 have been finally rejected under 35 U.S.C. §103(a) as allegedly unpatentable over *Lowry* in view of Barequet ("A data front-end for layered manufacturing", Annual Symposium on Computational Geometry; Proceedings of the 13th Annual Symposium on Computational Geometry, 1997; hereinafter "*Barequet*").

Claims 14, 16 and 17 are allowed.

Claims 1-13 are the subject of this appeal.

IV. STATUS OF AMENDMENTS

An amendment filed on December 16, 2003, in response to a Final Office Action mailed August 28, 2003 (Paper No. 21) and an Advisory Action mailed December 11, 2003 (Paper No. 24), has been entered for purposes of Appeal, according to an Advisory Action mailed January 12, 2004 (Paper No. 27).

V. SUMMARY OF THE INVENTION

To assist in understanding the invention, some background context is first provided.

It is not uncommon to find multiple software applications that perform similar functions but which are based on incompatible data formats. For example, the two software applications, Microsoft Word® and WordPerfect®, are both word-processing programs. They can each be used to create or generate similar documents. However, although the two programs can be used to perform similar functions, they each support, and therefore generate, documents or files using their own particular data format.

To resolve the problem of incompatible data formats, many programs include routines or functions that can be used to translate a document or file that was created by a first application ("source application"), into a document or file that has a data format that is supported by a second application ("target application"). For example, the software application Word® includes a set of software translation routines or functions that can translate a document or file that was created by WordPerfect® into a document or file that has a data format that is supported by Word®.

However, in translating from one ("source") format to another incompatible ("target") format, underlying data constructs associated with the respective formats may not directly map to each other. Thus, the "translated" copy of a document is effectively severed from the "original" copy of the document. Continuing with the example given above, a Word® document created by translating a WordPerfect® document exists entirely independently from the WordPerfect® document once the initial translation operation is completed.

This "separation" of the translated document from the original document has some significant consequences. For example, if changes are made, after the translation, to the original document, then the changes generally do not get propagated to the translated document. To force them into the translated document, one could perform a subsequent translation operation. However, subsequent translations will result in the loss of any intervening changes to the translated document, or might introduce duplicate data representations in the translated document.

For example, assume that a user makes some changes to a Word® document that was created by translating a WordPerfect® document. Assume further that, after making the changes to the Word® document, the user goes back and makes changes to the original WordPerfect®

document. The changes made to the original WordPerfect® document will not be reflected in the Word® document. If a subsequent translation is performed on the revised WordPerfect® document, the resulting Word® document will have lost the changes that had been made to the previous Word® document.

It should help to clarify the invention by noting that the term "object" is used in more than one context in the specification. For example, when referring to a "source object", a "target object", or "translation of an object", an "object" is data that is in a format dictated by an application. In this context, an object may be, for example, a Word® document or, in the example used most frequently in the specification, data that represents an entity within a computer aided design model, such as data that represents a car, a car door, or a surface that is part of a car door. This context contrasts with how the term "object" is used in object-oriented programming.

However, the specification does use the term "object" in a manner similar to the "object-oriented" sense of the term when describing a "filter object" or "collection object". Specifically, in this context, an object is a data structure or construct associated with computer programming or processing, such as but broader than, an object-oriented programming object.

Embodiments of the invention provide for a method for translating objects between applications that use different formats through use of a linking mechanism (302). The linking mechanism maintains a system for mapping each source object to a corresponding target object. Hence, upon an update to a source object in a source format using the source application, the linking mechanism is used to:

- apply the update to an object definition of a target object in the target application format,

- without removing intervening updates that may have been made to the target object using the target application, and
- without causing the introduction of unwanted duplicate objects in the target file.

One method described for mapping source objects to corresponding target objects includes generation of a hierarchical structure for organizing properties of a source object, such as an object represented in a computer aided design (CAD) application format, that is being translated to a target object, such as an object represented in a rendering application format, wherein each level of the hierarchy is associated with a property of corresponding objects. (Specification page 16, line 9 through page 17, line 2). Further, filter objects (432, 433, 434) are used to determine a location within the hierarchical structure to map source object properties, such as object identification, thickness, color, layer, etc., in the case of a CAD object. Thus, filter objects are associated with respective levels of the hierarchical structure and used to organize object properties into branches of the hierarchical structure, wherein the hierarchical structure is used to construct the target object in the target application. (Specification page 17, lines 3-16). Leaf objects (420, 421) are used to store, or cache, translated object properties, for use in constructing the target object. (Specification page 23, lines 12-17).

Collection objects (414-417) are used in conjunction with filter objects to compare and insert object properties into the hierarchical structure. A collection object either links to one or more collection objects or to a single leaf object. (Specification page 21, lines 19-22; page 22, lines 4-9).

Modifier stacks (422, 423, 430, 431) are used to record and maintain modifications that are made to objects through use of the target application. A modifier stack is associated with an object property, which is associated with a level of the hierarchical structure, and is linked with a

particular collection object. Hence, a modification associated with a given modifier stack is applied to the target file via the linked collection object to construct a target object. (Specification page 23, line 18 through page 24, line 8). To construct a target object, object properties that were translated and stored within each leaf object in the hierarchical structure are passed back up the hierarchical structure to a corresponding node object. As the properties are passed up the branches of the hierarchical structure, the modification that is stored in each of the modifier stacks is used to modify the properties of the objects. (Specification page 25, line 5 through page 26, line 16).

Filter objects are further used to determine at which level of the hierarchical structure modifications to an object made through use of the source application are stored. The modification of the property of the source object is applied to the target file at the determined level, and a modification associated with a modifier stack associated with the determined level is applied to the target file to construct the target object. Hence, according to this process, **any modifications made to an object property in either the source or target applications are preserved and properly applied to the target object.** (Specification page 25, line 5 through page 26, line 16).

VI. ISSUES

(A) Are Claims 1, 2, 4-10, 12 and 13 patentable under 35 U.S.C. §102(b) over Lowry et al. (U.S. Patent No. 4,864,497)?

(B) Are Claims 3 and 11 patentable under 35 U.S.C. §103(a) over *Lowry* in view of Barequet ("A data front-end for layered manufacturing"; "*Barequet*")?

VII. GROUPING OF CLAIMS

VIII. ARGUMENTS

A. THE LIMITATIONS OF CLAIMS 1, 2, 4-10, 12 AND 13 ARE NOT DISCLOSED IN THE *LOWRY* PATENT

With respect to Claim 1, before specifically describing distinctions between the claim limitations and the disclosure of *Lowry*, the following concept regarding the claim warrants general emphasis.

The source object is generated by the source application, and the target object is generated by translating the source object. Therefore, the source application indirectly dictates the construct or constitution of the target object. Hence, it would be inconsistent with the language of Claim 1 to equate Claim 1's "target object" with something that could not possibly be created by translating a "source object" created by a "source application".

For example, if the source object is data representing a car door in a format dictated by a CAD application, then the target object is data representing a car door in a format dictated by the target application. For another example in the context of a database, such as in *Lowry*, if the source object is a database record having fields A, B and C, then the target object is a record having fields A, B and C. It is important to note that, if the source object is a database record having fields A, B and C, it would be improper to equate the target object to a record having fields A, B, C and D, especially where field D is a field that is not supported by the source application.

LOWRY

The portions of *Lowry* that were relied upon in the Office Action (namely col. 11, lines 40-65; col. 31, lines 29-42; FIG. 5A and FIG. 5B) describe a system in which multiple applications may retrieve data from the same slave database, and store data into the same master database. FIG. 5A is probably the best illustration of this aspect of *Lowry's* system. *Lowry's* database appears to have a locking mechanism (including database controller DBC 2920 and a DBC lock 569) to prevent two applications from updating the same field. Finally, *Lowry* discloses retrieval programs 580 and 590 for retrieving data from the slave database 520 to the application programs, and converters 540 and 570 for converting data from the application programs prior to storing the data in the master database 510.

HYPOTHETICAL

Lowry does not disclose or suggest that one of the applications 530 and 560 is capable of making a modification that is not supported by the other application. However, during a telephonic interview with the Examiner, it was explained that even if *Lowry* did disclose that one of applications 530 and 560 was able to make a modification that is not supported by the other application, *Lowry* would still fail to satisfy the limitations of Claim 1.

To explain how *Lowry* does not satisfy Claim 1, repeated reference was made in the interview to a hypothetical in which the "source application" was a database application that does not support Chinese characters, and the "target application" was a database application that does support Chinese characters. In this hypothetical, the target application is able to make a modification (i.e. the insertion of Chinese characters into a field of a record) that the source application does not support.

In the following explanation, that same hypothetical shall be used to illustrate that Claim 1 cannot be satisfied by any reasonable reading of *Lowry*.

DISCUSSION OF CLAIM 1

As a preliminary matter, it should be noted that the source and target objects recited in Claim 1 are not the same object, because **the target object is translated from the source object**. Thus, while *Lowry* discloses that the presence of a lock may prohibit one application from writing to a shared resource while another application has a write lock to that resource, such a lock would not prevent one application from modifying the claimed source object while another application modified the target object. Thus, the source and target objects are, generally, corresponding resources in different applications, rather than a shared resource.

Claim 1 recites, among other things:

(1) **revising said target object** in said target application (2) **to reflect said second modification to said source object** (3) **without removing said first modification to said target object.**

It is long-settled that for a proper anticipation rejection, a reference must show each and every feature of a claim in the same combination as claimed. Connell v. Sears, Roebuck & Co., 722 F.2d 1542, 1548, 220 USPQ 193, 198 (Fed. Cir. 1983). Therefore, each of the limitations delineated as (1)-(3) must be disclosed in *Lowry* for *Lowry* to anticipate Claim 1. In the following discussion, it shall be explained that *Lowry* does not disclose or suggest the preceding limitations.

NO POSSIBLE SELECTION OF "SOURCE OBJECT" WILL CAUSE CLAIM 1 TO BE SATISFIED

The Office Action did not make clear what, in *Lowry*, was supposed to correspond to the "source object" and "target object" recited in the claims. During the telephonic interview, no specific answer was given to what, within *Lowry*, was considered to be the "source object" and the "target" object of the claims. Therefore, it was explained how Claim 1 was not satisfied regardless of what, within *Lowry*, was considered to be the "source object" and the "target object". The following is an attempt to reduce in writing that explanation.

Keeping in mind that the original source object dictates the constitution of the original translated target object by way of the translation, three possible "definitions" of the source object are hypothetically discussed, none of which will cause Claim 1 to be anticipated by *Lowry*. Specifically, scenarios are discussed, in the context of a database system as in *Lowry*, in which the source object is equated with:

(1) a field of a record;

(2) an entire record that includes a field that is modifiable and supported only by the target application; and

(3) an entire record except for a field that is modifiable and supported only by the target application.

For the purpose of explanation, reference shall be made to the scenario in which the target application supports use of Chinese characters, and the source application does not support the use of Chinese characters.

Scenario (1)

In scenario (1), suppose a particular field is considered the source object and such field is converted from a source application to the *Lowry* "common data structure" and converted from the common data structure to a corresponding field in a target application (target object). Assume that the target application obtains a lock on that field in the target object, modifies that field by adding Chinese characters, and converts and saves the field back to the *Lowry* database.

At this point, if the source application modifies that field of the original source record (perhaps from its local store) and converts and saves it back to the *Lowry* database, (a) the Chinese characters will be overwritten or (b) another separate record, or snapshot, will be created separate from the record or snapshot containing the field with the Chinese characters. Thus, this scenario could not possibly satisfy the limitations:

(1) **revising said target object** in said target application (2) **to reflect said second modification to said source object** (3) **without removing said first modification to said target object.**

Lowry does not disclose a target object being revised to contain both modifications to corresponding source and target objects. Even if the source application retrieved the target object/field from the *Lowry* database to perform its modification on that object/field, the Chinese characters would still be lost upon conversion to the source application because, by definition, the source application does not support Chinese characters. Thus, the target object will never reflect both the Chinese characters, and the subsequent change made by the source application to the source object.

Scenario (2)

Scenario 2 has a faulty premise, because it does not make sense to say that the source object is a record with a field that is modifiable and supported only by the target application. This is not possible, because the source object is generated by the source application, and it makes no sense to say that an application generates and modifies an object on the one hand, but does not support the object on the other hand.

In scenario (2), the source object coming from the source application cannot be a record with a field that is modifiable and supported only by the target application (in other words, not supported by the source application), as required by Claim 1. Thus, this scenario is not feasible by virtue of the limitation that **the modification made to the target object is not supported by the source application**. The construct of the original source object that is assumed in this scenario, from which the target object is translated, is just not possible in a database record context such as in *Lowry*, when constrained to the foregoing limitation.

Furthermore, even if the modification made by the source object were to a version stored in the *Lowry* database which included the field to which Chinese characters were added (i.e., the target object, contrary to the claim language regarding the "second modification to said source object in said source application"), those characters would be lost when the object is converted from the common data structure of *Lowry* to the source application prior to making the modification by the source application and converting and storing back into the common data structure. *Lowry* does not teach or suggest a way to insert the Chinese characters back into the target object, now corresponding to the source-modified version of the object, at the target application so that the target object contains the modifications to the corresponding objects from both applications.

Scenario (3)

In scenario (3), the original source object coming from the source application does not include the field that is modifiable and supported only by the target application. However, if the original source object coming from the source application does not include the field that is modifiable and supported only by the target application, then the translated target object that corresponds to such a source object will not have that field either. Thus, the target application is unable to modify a field that doesn't exist, such as a field of a data type that supports Chinese characters. *Lowry* does not teach or suggest a way to insert a new field into a record, which is supported only by the target application, so that the target object is revised to contain the modifications to the corresponding objects from both applications.

To summarize the inconsistency of the foregoing hypothetical scenarios, regardless of what resource is chosen from the database of *Lowry* to represent the source object in Claim 1, *Lowry* does not disclose or suggest all of the limitations of the claim. Therefore, *Lowry* cannot and does not anticipate Claim 1.

Furthermore, regarding the "snapshot file 525" of *Lowry*, col. 11, lines 52-55 states that the snapshot file is designed to represent a plurality of versions of master data file 515. Hence, if the *Lowry* database were used in an attempt to capture both the modification made by the source application and the modification made by the target application, there would be no single version of the master file that includes both of the modifications. Rather, there might be two different versions: one represented in a first snapshot file containing the first modification and one represented in a second snapshot file containing the second modification, with no way of revising the target object to reflect both modifications.

A LOCK MECHANISM ON A SHARED RESOURECE DOES NOT SATISFY THE LIMITATIONS OF CLAIM 1

Subsequent to presenting the foregoing explanation, some of the rejections were maintained based on a misunderstanding of one of the scenarios. Specifically, the rejection relied on the disclosure of a data lock mechanism associated with a database of shared (among more than one application) resources to support the current rejection of Claim 1. The following discussion addresses that misunderstanding, to overcome the rejection based thereon.

The Office Action contended that Applicants' remarks regarding the first scenario (scenario 1) of the previously-presented hypothetical are based on an assumption that "the lock on the field in the target object is released before the source application modifies that object." As is explained hereafter, the first scenario makes no such assumption.

The Office Action goes on to say that releasing the lock removes the overwrite protection and that "[i]f the target lock is not released, then the source application is not able to overwrite the changes made by the target application" and, therefore, a scenario in which the lock is maintained satisfies the claimed limitations. As is also explained hereafter, whether or not the lock is released or maintained, the limitations of Claim 1 would not be satisfied.

Specifically, regardless of whether the lock of *Lowry*, as applied by the Office Action to Claim 1, is released or not, Lowry does not anticipate Claim 1. To reiterate, Claim 1 recites, among other things:

> (1) **revising said target object** in said target application (2) **to reflect said second modification to said source object** (3) **without removing said first modification to said target object.**

If the hypothetical lock of *Lowry* was not released, then limitations (1) and (2) are not satisfied. In other words, if a locking mechanism is used in an attempt to replicate Claim 1 as alleged in the Office Action, with a lock being held on the target object by the target application against the source application, then the **second modification to the source object, which is reflected in the target object**, would not be completed, as explicitly recited in Claim 1.

On the other hand, if the lock is released, then limitations (1) and (3) are not satisfied. That is to say, the **first modification to the target object would be removed by reflecting the second modification in the target object**. The Office Action is correct in recognizing that release of the lock would allow the source application to **overwrite** the object. However, as anyone who uses a word processor has painfully discovered, when something is overwritten, previous changes to the file are lost. This is in direct conflict with an explicit limitation of Claim 1.

If the source application modifies a field of the original source record and converts and saves it back to the *Lowry* database, (a) the modification made by the target application will be overwritten, or (b) a new record, or snapshot, will be created with the modification made by the source application, separate from the record or snapshot containing the field with the modification made by the target application. Thus, this scenario could not possibly satisfy the limitations recited in Claim 1. In fact, the analyses presented in the Office Actions, when refuted, further exemplify some of the advantages and value of the invention recited in Claim 1, for they show that a simple locking structure cannot be used to accomplish what is accomplished by the invention recited in Claim 1.

Based on the foregoing discussion, it is respectfully submitted that *Lowry* does not anticipate Claim 1 under 35 U.S.C. §102(b) because it does not teach or disclose the limitations

of Claim 1. In summary, a common database and associated data structure for resources that are shared among multiple applications, in conjunction with a locking mechanism on such shared resources, does not anticipate the invention recited in Claim 1.

Claims 2-8 depend directly or indirectly from Claim 1. Therefore, Claims 2-8 are patentable over the references of record for at least the same reasons as Claim 1. Hence, for at least the foregoing reasons discussed in reference to Claim 1, Claims 2 and 4-8 are also not anticipated by *Lowry*.

DISCUSSION OF CLAIM 9

Independent Claim 9 recites limitations similar to Claim 1, whereby, generally, a sequence of modifications are made to respective corresponding objects in different applications that support different formats and both the modifications are reflected in a single object (the "second object"). Therefore, reasons for patentability based on distinguishing over *Lowry* as discussed above in reference to Claim 1 are also reasons for the patentability of Claim 9, to the extent applicable.

Claims 10 and 11 depend from Claim 9. Therefore, Claims 10 and 11 are patentable over the references of record for at least the same reasons as Claim 9.

DISCUSSION OF CLAIMS 12 AND 13

Claims 12 and 13 are computer-readable medium and system claims that correspond to the method of Claim 1. Hence, Claims 12 and 13 are patentable over the cited references of record for at least the same reasons as Claim 1.

RESPONSE TO EXAMINER'S COMMENTS

In paragraph 35 of the Office Action, the Examiner contends that Applicant/Appellant is arguing features that are not in the Claim 1, when noting that the target object is translated from

the source object and, therefore, that the target and source objects are corresponding resources rather than a shared resource. This allegation is misguided, in that Claim 1 explicitly recites that the source object is generated by a source application and that the target object is **translated**, in a target application, **from the source object.** Therefore, it is clear by the plain meaning of the claim language that the source object and the target object are different, but corresponding, objects, and any contrary interpretation of the claim language is an incorrect and unsupported interpretation. Appellant acknowledges that Examiner is entitled to give the broadest reasonable interpretation to the claim language. However, interpreting the source object and target object to be a shared resource is not reasonable in light of the claim language.

Furthermore, in paragraph 35 of the Office Action, the Examiner contends that Applicant/Appellant is arguing features that are not in the Claim 1, with respect to whether the steps recited in Claim 1 are performed simultaneously or sequentially. This allegation is misguided, in that Appellant provides no discussion as to whether the steps of Claim 1 are limited to sequential performance or simultaneous performance. An interpretation of the plain meaning of the claim language shows that the only such inherent temporal limitation is that the first modification and the second modification are performed prior to the revising step.

The remainder of the Examiner's comments with respect to Claims 1-13, to the extent necessary, are addressed and refuted throughout this brief.

B. THE LIMITATIONS OF CLAIMS 3 AND 11 ARE NOT TAUGHT, DISCLOSED OR SUGGESTED IN THE *LOWRY* AND *BAREQUET* REFERENCES

Claim 3 depends from Claim 1 and Claim 11 depends from Claim 9. *Lowry* is relied on in the rejection for the limitations of the independent Claims 1 and 9, however, it is shown above that *Lowry* does not disclose or suggest several limitations of these claims. Therefore, *Lowry*

does not support a prima facie case of obviousness with respect to Claims 3 and 11. In addition, *Baraquet* does not cure the deficiencies of *Lowry*. Since both of Claims 1 and 9 are patentable over *Lowry*, it follows that dependent Claims 3 and 11 are patentable over *Lowry* in view of *Baraquet*, when *Lowry* is relied on for the same limitations as with the rejection of the parent claims. For these reasons, Claims 3 and 11 are patentable over the cited references of record.

CONCLUSION AND PRAYER FOR RELIEF

Based on the foregoing, it is respectfully submitted that the rejections of (A) Claims 1, 4-10, 12 and 13 under 35 U.S.C. §102(b); and (B) Claims 3 and 11 under 35 U.S.C. §103(a), lack the requisite factual and legal bases. Appellant therefore respectfully requests that the Honorable Board reverse the respective rejections of Claims 1-13.

Respectfully submitted,

HICKMAN PALERMO TRUONG & BECKER LLP

Date: 1/19/04

John D. Henkhaus
Registration No. 42,656

1600 Willow Street
San Jose, California 95125-5106
Tel: (408) 414-1203
Fax: (408) 414-1076

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, Mail Stop - Appeal Brief Patents, Box 1450, Alexandria, VA 22313-1450

on 1/20/04 by Clare Fung

IX. APPENDIX

1. A method for translating objects between applications that use different formats, the method comprising:

generating a source object in a source application;

translating the source object to a target object in a target application, wherein the target application has a format that is not supported by the source application;

performing a first modification to the target object, wherein said first modification is not supported by said source application;

performing a second modification to said source object in said source application; and

revising said target object in said target application to reflect said second modification to said source object without removing said first modification to said target object.

2. The method of Claim 1, wherein the step of performing the first modification to the target object includes the step of performing a type of modification that cannot be performed using said source application.

3. The method of Claim 1, wherein:

the source application is a Computer Aided Design (CAD) application;

the target application is a rendering application; and wherein

the step of generating the source object in the source application includes the step of generating a CAD object in said CAD application;

the step of translating the source object to the target object includes the step of translating the CAD object into a rendering object;

the step of performing the first modification to the target object includes the step of performing a modification to the rendering object;

the step of performing a second modification to said source object includes the step of performing a modification to the CAD object; and

the step of revising said target object includes the step of revising the rendering object to reflect the second modification that was made to the CAD object without undoing the first modification to the rendering object.

4. The method of Claim 1, wherein:

the source object is associated with a source geometry and one or more source properties; and

the step of translating the source object to the target object includes the steps of translating the source geometry to a target geometry; and

translating the one or more source properties to one or more target properties.

5. The method of Claim 1, wherein the step of translating the source object to the target object includes the step of:

building a mapping based on a translation between the source object and the target object.

6. The method of Claim 5, wherein the step of building the mapping includes the step of:

constructing a hierarchical tree structure, wherein the hierarchical tree structure is based on one or more properties associated with the source object.

7. The method of Claim 6, wherein

the source object is associated with a source geometry and one or more source properties; and

the step of constructing the hierarchical tree structure includes the steps of:

generating a set of tree objects, wherein the set of tree objects include one or more filter objects that are based on said source properties;

translating the source geometry to a target geometry; and

inserting said target geometry into said hierarchical tree structure based said one or more filter objects.

8. The method of Claim 7, wherein the step of generating the set of tree objects includes the steps of:

translating the one or more source properties to one or more target properties;

generating one or more modifier stacks, wherein the one or more modifier stacks are based on the one or more target properties; and

inserting the one or more modifier stacks into the hierarchical tree structure.

9. A method for translating objects between applications that use different formats, the method comprising:

generating a first object in a first application;

translating the first object to a second object in a second application, wherein the second object has a format that is not supported by the first application;

performing a first modification to the second object in the second application;

performing a second modification to said first object in said first application; and

performing a third modification to the second object based on data generated in response to said second modification to said first object, wherein said third modification causes said second object to reflect the second modification that was made to the first object without undoing the first modification to the second object.

10. The method of Claim 9, wherein the step of performing the first modification to the second object includes the step of performing a type of modification that cannot be performed using said first application.

11. The method of Claim 9, wherein:

the first application is a Computer Aided Design (CAD) application;

the second application is a rendering application; and wherein

the step of generating the first object in the first application includes the step of generating a CAD object in said CAD application;

the step of translating the first object to the second object includes the step of translating the CAD object into a rendering object;

the step of performing the first modification to the second object includes the step of performing a modification to the rendering object;

the step of performing a second modification to said first object includes the step of performing a modification to the CAD object; and

the step of performing the third modification to the second object includes the step of performing a third modification to the rendering object to reflect the second

modification that was made to the CAD object without undoing the first modification to the rendering object.

12. A computer-readable medium carrying one or more sequences of instructions for translating objects between applications that use different formats, wherein execution of the one or more sequences of instructions by one or more processors causes the one or more processors to perform the steps of:

generating a source object in a source application;

translating the source object to a target object in a target application, wherein the target application has a format that is not supported by the source application;

performing a first modification to the target object, wherein said first modification is not supported by said source application;

performing a second modification to said source object in said source application; and

revising said target object in said target application to reflect said second modification to said source object without removing said first modification to said target object.

13. A system for translating objects between applications that use different formats, the system comprising:

a memory;

one or more processors coupled to the memory; and

a set of computer instructions contained in the memory, the set of computer instruction including computer instructions which when executed by the one or more processors, cause the one or more processors to perform the steps of:

generating a source object in a source application;

translating the source object to a target object in a target application, wherein the target application has a format that is not supported by the source application;

performing a first modification to the target object, wherein said first modification is not supported by said source application;

performing a second modification to said source object in said source application; and

revising said target object in said target application to reflect said second modification to said source object without removing said first modification to said target object.